BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Perangkat Lunak

Implementasi perangkat lunak bertujuan untuk memastikan perangkat lunak yang dibangun dapat bekerja baik dan sesuai dengan tujuan dibangun. Sebelum perangkat lunak diimplementasikan, maka perangkat lunak harus bebas dari kesalahan. Kesalahan yang mungkin terjadi antara lain kesalahan tampilan ataupun kesalahan proses. Setelah perangkat lunak bebas dari kesalahan, kemudian dapat dilakukan pengujian dengan menjalankan perangkat lunak.

4.1.1 Implementasi Proses

Proses-proses yang terdapat dalam sistem ini adalah sebagai berikut:

a. Proses Pengambilan data GPS

Proses pengambilan data dari GPS dilakukan dengan melakukan import *library* "*Core Location*", kemudian mengimplementasi *method* "*didUpdateToLocation*" untuk mendapatkan titik koordinat terakhir. Setelah mendapatkan titik koordinat maka dapat diketahui nilai bujur, lintang, dan ketinggian lokasi. Kemudian tiga nilai tersebut disimpan dalam suatu *array* yang nantinya akan digunakan untuk menghitung waktu sholat.

b. Proses Perhitungan Waktu Sholat

Proses perhitungan waktu sholat dibagi lagi menjadi enam proses, yaitu:

1. Proses perhitungan *equation of time*

Proses perhitungan ini adalah proses yang paling penting dalam perhitungan waktu sholat. Proses ini mengambil data dari *array* yang telah disediakan oleh proses pengambilan data GPS, atau dari data penyimpanan

daftar kota jika GPS tidak tersedia. Setelah mendapatkan data tersebut, proses ini menghitung nilai *equation of time* yang telah dibagi menjadi beberapa fungsi dalam pemrogramannya. Kode pemrograman dari proses ini adalah sebagai berikut:

```
-(double) hitungJulianDay:(int)tgl:(int)bln:(int)thn{
      float JD;
      if(bln == 1 || bln == 2){
            bln = bln + 12;
            thn = thn - 1;
      }
      int A = (int) thn/100;
      int B = 2 + (int) ((A/4) - A);

JD = 1720994.5 + ((int) (365.25*thn)) + ((int) (30.6001 * (bln
      + 1))) + B + tgl + (12/24.0);

      return JD;
}

-(double) hitungJulianDayLocal:(double) JD_GMT:(int) z{
      double JDLocal = JD_GMT - (z/24.0);

      return JDLocal;
}

-(double) hitungEquationOfTime:(double) JD_Local{
      double U = (JD_Local - 2451545)/36525.0;

double L0 = ((280.46607 + (36000.7698 * U)) * PHI / 180.0);

double EOT = ((-1*(1789 + 237 * U))*sin(L0) - (7146 -
      62*U)*cos(L0) + (9934 - 14*U)*sin(2*L0) - (29 +
      5*U)*cos(2*L0) + (74 + 10*U)*sin(3*L0) + (320 -
      4*U)*cos(3*L0) - 212*sin(4*L0))/1000.0;
      return EOT;
}
```

2. Proses perhitungan waktu sholat Dhuhur

Proses berikutnya adalah menghitung waktu sholat Dhuhur dengan menggunakan rumus yang ada. Hasil dari perhitungan *equation of time* digunakan dalam perhitungan ini. Kode pemrograman untuk proses ini adalah sebagai berikut:

```
      -(double) hitungWaktuDhuhur{
double JDGMT = [self hitungJulianDay: self.tanggal: self.bulan:
      self.tahun];
double JDLocal = [self hitungJulianDayLocal: JDGMT: self.zona];
      double EOTime = [self hitungEquationOfTime: JDLocal];
      double B = self.bujur;

      double dhuhur = 12 + self.zona - B/15 - EOTime/60;

return dhuhur;
}
```

3. Proses perhitungan waktu sholat Ashar

Proses perhitungan waktu sholat Ashar dipengaruhi oleh nilai dari hasil perhitungan waktu sholat Dhuhur, dan juga dipengaruhi oleh nilai tetapan sholat Ashar yang ditentukan oleh beberapa mazhab. Sebelum menghitung waktu sholat Ashar, proses ini terlebih dahulu menghitung *hour angle* untuk dijadikan salah satu parameter perhitungan. Kode pemrograman yang digunakan untuk menghitung waktu sholat Ashar adalah sebagai berikut:

```
      -(double) hitungDeklinasiMatahari:(double) JD_Local{
      double T = 2 * PHI * (JD_Local - 2451545) / 365.25;
double Delta = 0.37877 + 23.264 * sin((57.297 * T - 79.547) *
      PHI /180.0)+0.3812 * sin((2 * 57.297 * T - 82.682) *
      PHI/180.0)+0.17132*sin((3*57.297*T-59.722)*PHI/180.0);
      double Delta_radian = Delta * PHI / 180.0;
```

```
        return Delta_radian;
}

-(double) hitungHourAngle:(double) altitude{
        double lintangRad = self.lintang * PHI / 180;
double JDGMT = [self hitungJulianDay: self.tanggal: self.bulan:
        self.tahun];
double JDLocal = [self hitungJulianDayLocal: JDGMT: self.zona];
double Delta = [self hitungDeklinasiMatahari: JDLocal];
double CosHA = (sin(altitude) - sin(Delta) * sin(lintangRad)) /
        (cos(Delta)*cos(lintangRad));
        double HA = acos(CosHA)*180/PHI;

        return HA;
}

-(double) hitungWaktuAshar:(double)dhuhur{
        double lintangRad = self.lintang * PHI / 180;
double JDGMT = [self hitungJulianDay: self.tanggal: self.bulan:
        self.tahun];

double JDLocal = [self hitungJulianDayLocal: JDGMT: self.zona];

double Delta = [self hitungDeklinasiMatahari: JDLocal];

double      altitude      =      atan(1/([self      KA]+tan([self
        absolute:(lintangRad-Delta)])));

        double HAAshar = [self hitungHourAngle:altitude];

        double ashar = dhuhur + (HAAshar / 15);

        return ashar;
}
```

4. Proses perhitungan waktu sholat Maghrib

Proses perhitungan waktu Maghrib dipengaruhi oleh nilai dari hasil perhitungan waktu sholat Dhuhur, dan juga dipengaruhi oleh ketinggian suatu lokasi. Kode pemrograman untuk proses ini adalah sebagai berikut:

```
-(double) hitungWaktuMaghrib:(double) dhuhur{
double altitude = (-0.8333-0.0347*(sqrt([self
ketinggian])))*PHI/180; //udah di radian
double HAMaghrib = [self hitungHourAngle:altitude];
double maghrib = dhuhur + (HAMaghrib / 15);

return maghrib;
}
```

5. Proses perhitungan waktu sholat Isya'

Proses perhitungan waktu sholat Isya' dipengaruhi oleh nilai dari hasil perhitungan waktu sholat Dhuhur, dan juga dipengaruhi oleh nilai sudut Isya'. Kode pemrograman dari proses perhitungan waktu sholat Isya' ini adalah sebagai berikut:

```
-(double) hitungWaktuIsya:(double) dhuhur{
double altitude = -([self sudutIsya])*PHI/180;
double HAIsya = [self hitungHourAngle:altitude];
double isya = dhuhur + (HAIsya / 15);

return isya;
}
```

6. Proses perhitungan waktu sholat Subuh

Proses perhitungan waktu sholat Subuh dipengaruhi oleh hasil perhitungan waktu sholat Dhuhur, dan juga dipengaruhi oleh nilai dari sudut Subuh. Kode pemrograman dari proses ini adalah sebagai berikut:

```
-(double) hitungWaktuSubuh:(double) dhuhur{
```

```
    double altitude = -([self sudutSubuh])*PHI/180;
    double HASubuh = [self hitungHourAngle:altitude];
    double subuh = dhuhur - (HASubuh / 15);

    return subuh;
}
```

Setelah melakukan berbagai perhitungan waktu sholat, kemudian sistem akan menampilkan hasil dari perhitungan tersebut. Hasil ditampilkan dengan menggunakan objek dari kelas *UITableView* agar tampilan menjadi rapi. *UITableView* adalah kelas standar yang digunakan dalam pemrograman *iPhone* untuk menampilkan data seperti tabel satu kolom dan dapat dikonfigurasi sesuai dengan keinginan. *UITableView* juga dipakai oleh berbagai aplikasi-aplikasi standar *iPhone* seperti *iPod, settings, YouTube, Photos*, dan sebagainya (Apple Inc., 2009:212). Adapun hasil yang ditampilkan oleh sistem adalah seperti pada Gambar 4.1.

Gambar 4.1 Halaman lihat jadwal sholat

c. Proses Perhitungan Arah Kiblat

Proses perhitungan arah kiblat dilakukan dengan cara mengambil pengaturan lintang dan bujur tempat, kemudian menghitung arah kiblat dengan

rumus yang ada. Kode pemrograman yang digunakan dalam proses ini adalah sebagai berikut:

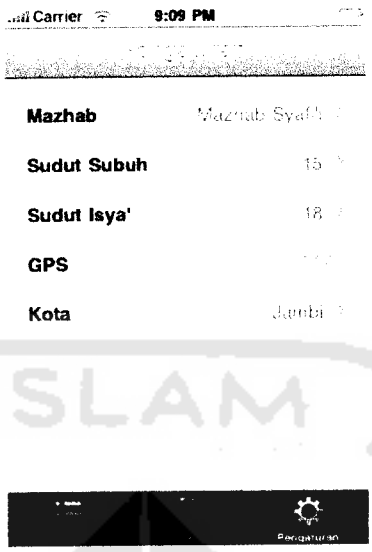
```
-(double) hitungArahKiblat{
      double BK = BujurKabah * PHI / 180;
      double BT = self.bujur * PHI / 180;
      double LK = LintangKabah * PHI / 180;
      double LT = self.lintang * PHI / 180;

      double    tanK    =    sin(BK-BT)    /    (cos(LT)*tan(LK)-
            sin(LT)*cos(BT-BK));

      double K_radian = atan(tanK);

      double K = K_radian * 180 / PHI;
      if(K<0){
            K+=360;
      }

      return K;
}

-(double) hitungJarakKabah{
      double BK = BujurKabah * PHI / 180;
      double BT = self.bujur * PHI / 180;
      double LK = LintangKabah * PHI / 180;
      double LT = self.lintang * PHI / 180;

      double cosD = sin(LT)*sin(LK) + cos(LT)*cos(LK)*cos(BT -
BK);
      double D = acos(cosD);
      double jarak = 6378.137 * D;

      return jarak;
}
```

Setelah melakukan perhitungan tersebut, proses selanjutnya adalah menampilkannya kepada *user*. Tampilan dari proses ini adalah seperti pada Gambar 4.2.

Gambar 4.2 Halaman lihat arah kiblat

d. Proses Ganti Pengaturan

Proses ganti pengaturan adalah sebuah proses untuk menampilkan form-form pengaturan, kemudian menerima hasil pengaturan yang dimasukkan oleh *user*, dan menyimpannya kedalam file penyimpanan pada sistem. *UITableView* digunakan untuk menampilkan menu pengaturan, dan *UINavigationController* digunakan untuk mengatur menu-menu tersebut kedalam sebuah hirarki yang rapi. Tampilan dari halaman utama ganti pengaturan adalah seperti pada Gambar 4.3.



Gambar 4.3 Halaman ganti pengaturan

4.2 Analisis Kinerja Perangkat Lunak

Analisis kinerja perangkat lunak merupakan tahap dimana sistem dioperasikan pada tahap yang sebenarnya, sehingga akan diketahui apakah sistem yang telah dibuat benar-benar sesuai dengan yang direncanakan. Pada bagian ini juga dilakukan beberapa pengujian untuk penanganan kesalahan yang terdapat dalam aplikasi.

4.2.1 Pengujian Normal

Pengujian normal dilakukan dengan cara memberikan masukan pada aplikasi. Berikut beberapa pengujian yang dilakukan pada aplikasi penentuan waktu sholat.

a. Pengujian Perhitungan Jadwal Sholat

Jadwal sholat yang akan diujikan adalah pada tanggal 7 Agustus 2010, dengan menggunakan mazhab syafi'i. Kemudian nilai sudut subuh yang digunakan adalah 20 derajat, dan sudut isya yang digunakan adalah 18 derajat. Pengujian tidak bisa dilakukan menggunakan fitur GPS dalam aplikasi

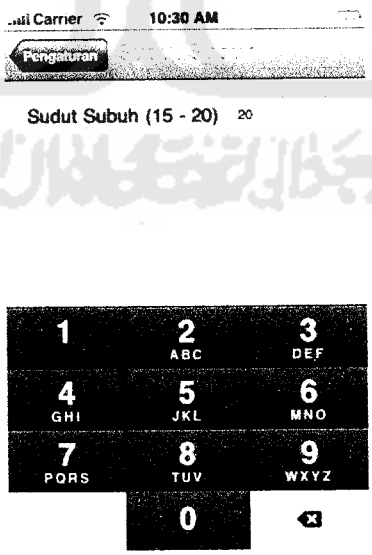
dikarenakan *iPhone Simulator* tidak terdapat perangkat GPS. Oleh karena itu, pengujian dilakukan dengan memilih satu kota, yaitu kota Jakarta.

Berikut adalah tampilan dari pengaturan yang disesuaikan dengan studi kasus diatas:

1. Pengaturan mazhab (lihat Gambar 4.4)

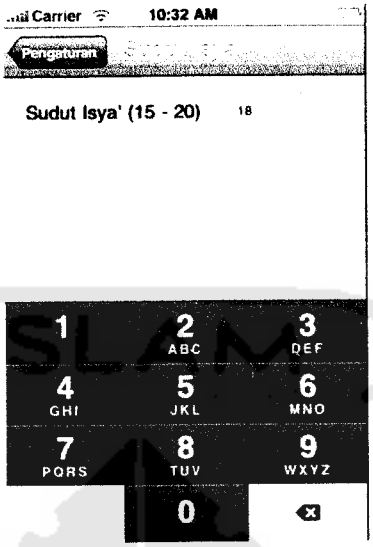
Gambar 4.4 Pengaturan mazhab

2. Pengaturan sudut subuh (lihat Gambar 4.5)



Gambar 4.5 Pengaturan sudut subuh

3. Pengaturan sudut isya (lihat Gambar 4.6)

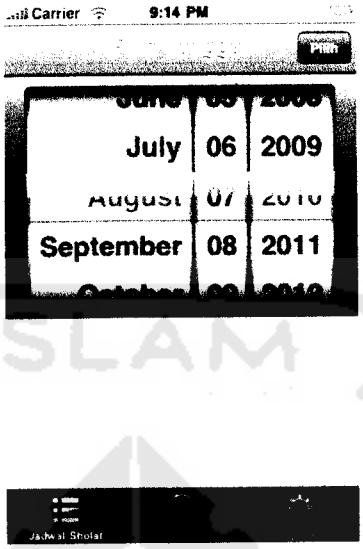


Gambar 4.6 Pengaturan sudut isya

4. Pengaturan kota (lihat Gambar 4.7)

Gambar 4.7 Pengaturan kota

5. Pengaturan tanggal (lihat Gambar 4.8)



Gambar 4.8 Pengaturan tanggal

Dari pengaturan tersebut, didapatkan hasil perhitungan jadwal sholat sebagai berikut (lihat Gambar 4.9).

Gambar 4.9 Hasil perhitungan jadwal sholat

Hasil dari perhitungan tersebut dibandingkan dengan hasil perhitungan dari aplikasi WinHisab yang dibuat oleh Departemen Agama Republik Indonesia (Badan Hisab dan Rukyat Departemen Agama Republik Indonesia, 1996), dan juga data dari Jakarta Islamic Centre (Jakarta Islamic Centre, 2010). Perhitungan menggunakan parameter yang sama untuk setiap sumber perhitungan, dan didapatkan hasil perbandingan seperti pada Tabel 4.1.

Tabel 4.1 Tabel Perbandingan Hasil Perhitungan Waktu Sholat Pada Tanggal 7 Agustus 2010

	Subuh	Dhuhur	Ashar	Maghrib	Isya
Aplikasi Penentuan Waktu Sholat	4:42	11:58	15:20	17:57	19:06
WinHisab Departemen Agama RI	4:44	12:01	15:22	17:57	19:08
Jakarta Islamic Centre	4:51	11:58	15:20	17:55	19:06

Dari hasil perbandingan, didapatkan selisih kurang lebih tiga menit untuk setiap waktu sholat, maka dapat disimpulkan bahwa perhitungan yang diperoleh dari aplikasi penentuan waktu sholat ini cukup akurat.

b. Pengujian Perhitungan Arah Kiblat

Perhitungan arah kiblat yang akan diujikan adalah pada kota Yogyakarta, Aceh, dan San Francisco. Hasil dari perhitungan arah kiblat tersebut dibandingkan dengan hasil perhitungan dari aplikasi Qibla Locator (Qibla Locator, 2010). Perhitungan menggunakan parameter yang sama dan didapatkan hasil perhitungan seperti pada Tabel 4.2.

Tabel 4.2 Tabel Perbandingan Hasil Perhitungan Arah Kiblat

	Yogyakarta		Aceh		San Francisco	
	Arah Kiblat	Jarak	Arah Kiblat	Jarak	Arah Kiblat	Jarak
Aplikasi Penentuan waktu sholat	294,71°	8356,28 km	292,59°	6504,74 km	19,32°	13225,89 km
Qibla Locator	294,71°	8357 km	292,25°	6259 km	19,26°	13222 km

Dari hasil perbandingan, didapatkan nilai yang sama, maka dapat disimpulkan bahwa perhitungan arah kiblat yang diperoleh dari aplikasi ini cukup akurat.

4.2.2 Pengujian Tidak Normal

Pengujian tidak normal adalah dengan memasukkan konfigurasi yang tidak sesuai dengan keadaan pada umumnya. Pengujian yang dilakukan adalah sebagai berikut:

a. Penanganan Kesalahan Masukan Data Kosong

Penanganan kesalahan ini dilakukan untuk menangkap setiap kesalahan yang terjadi ketika *field* pada pengaturan isian data kosong atau tidak sesuai dengan rentang nilai yang diperbolehkan. Contoh penanganan kesalahan ini terdapat pada pengaturan sudut Subuh dan Isya'. Jika nilai dari sudut dikosongkan, maka akan muncul peringatan seperti pada Gambar 4.10.

Gambar 4.10 Penanganan kesalahan masukan data kosong