**Insurance Supermarket Manual**

**Overview**

**Insurance Supermarket** is just a rough title, another can be chosen. The overall flow is this; the supermarket brings together clients (consumers of insurance products), insurance providers, and claim investigators. The providers and clients sign up but the claim investigators are added by providers. The flow is;

1. Provider signs up by clicking the *Insurance Providers* link in the home page.
2. Provider then logs in after sign up.
3. Provider posts policies – go to policies then links for all operations are there.
4. Provider adds claim investigators.
5. Client signs up (in the homepage) then logs in.
6. Client views policies posted by various providers, then buys policies.
7. If the policy is for Car, client povides more information about the car(s). All these under *policies->my policies*
8. Provider approved (or rejects) the request for policy. Client gets this feedback notification via email.
9. If approved, client pays for the cover.
10. Client can then file claim.
11. Claim investigator adds their report to the claim – client receives email update.
12. Provider enters final verdict on claim – client receives email update.
13. Provider can pay the claim by entering payment information on a form (no actual payment) – client receives email notification.
14. Client can observe from their end how claim is progressing by looking at claim status.
15. 7 days before expiry of a policy, clients get notifications. Now code runs every minute but in real-world that will every day.

It is a one-stop shop so clients get all insurance services in one place; talk of convenience and the power to choose their preferred provider.

**Structure**

Use the data dictionary pdf file attached to see the tables in the db representing various entities, and their relations. I have also attached in ERD but it is not clear. You’ll go to phpmyadmin, open the db there, go to *More* then click *Designer.* There you’ll see the relations in a clearer manner. The naming of foreign keys will also help you to identify the tables they relate.

The framework used is the php Yii framework version 1.1. To get it up and running, do the following;

1. Extract the file **YiiNew** in htdocs or www (assuming that you have xampp, or wampp installed).
2. Then extract the **insurance** file as well.
3. On your browser you’ll type *127.0.0.1/insurance/index.php* (due to time constraints, I’ve not been able to work on removing the index.php but works fine with it). Bingo.
4. Remember to import the db **insurance.sql** to your PhpMyAdmin.
5. Go to *insurance/protected/config/main.php* where you’ll see some options to change, the main one being the database under the **db** array.

**General Usage**

For code manipulation we have model, view and controller classes at the same directory level as the **config** folder – those are for the client. Then under **modules** we have the investigator and provider. You can let me know if you want to make changes to any of the codes so that I don’t type a very big document here, haha. But you’ll brush through to see how it is.

When you go to **themes** folder under **redruby** or **citylights** then you proceed to **layouts** then **main.php**, you’ll see a jQuery function known as **expiryChecker().** That’s what runs after every minute to send expiry notificaions via email, you can change the times – it is in miliseconds

For usage, you can create new users or go to the db and use the username and passwords that exists for the three categories of users – check the **clients** table, **provider** table, and the **investigators** table.

For better results, change the client emails to one that you can access to see the email notifications.

For payments we will not be able to implement M-Pesa fully now due to time constraints. Just inform the supervisor that that is how it is supposed to work if M-Pesa is enabled. For now users will just be typing the M-Pesa receipt number, assumably after sending money via M-Pesa.

*Do the manipulations, brush through it then in case of any clarification or direction I’ll be glad to be of service…*