Project Setup(Mac)

I have used beego MVC framework and visual studio code editor for the development of this project. You can setup the project by following way:

- In the attached zip file, you will find a mysql db backup file named ad_server_db.sql. Actually, It contains all the necessary statements to create db tables and insert some dummy data.
- Install golang from here: https://golang.org/dl/.
    - After installing go please set the gopath by following way:
    - Open bash_profile using: vi ~/.bash_profile
    - Then add following lines:
        - export GOROOT=/usr/local/go
        - export PATH=$GOROOT/bin:$PATH
        - export GOPATH=/Users/{your_user_name}/work (In my case it's 'rakib'. This will be the work directory)
        - export PATH=$GOPATH/bin:$PATH
- You will need to install Beego and the Bee dev tool. Run the following commands from the terminal to install the beego:
    - $ go get github.com/astaxie/beego
    - $ go get github.com/beego/bee
- Now unzip my attached zip file and copy ad-server folder. Then go to `cd $GOPATH/src` directory and paste my 'ad-server' folder.
- Go to `cd $GOPATH/src/ad-server` directory
- Before you run the project, please change the mysql connection string which will be found in main.go file at line #13.
    - orm.RegisterDataBase("default", "mysql", "username:password@/ad_server_db?charset=utf8")
- Execute following command in the terminal to run the solution:
    - `bee run`
- It will run the project at HTTP port 8080
- Open any browser and then hit this url http://localhost:8080 to see the running ad-server.
- In the HOME page, i have briefly described about all 3 API'S.

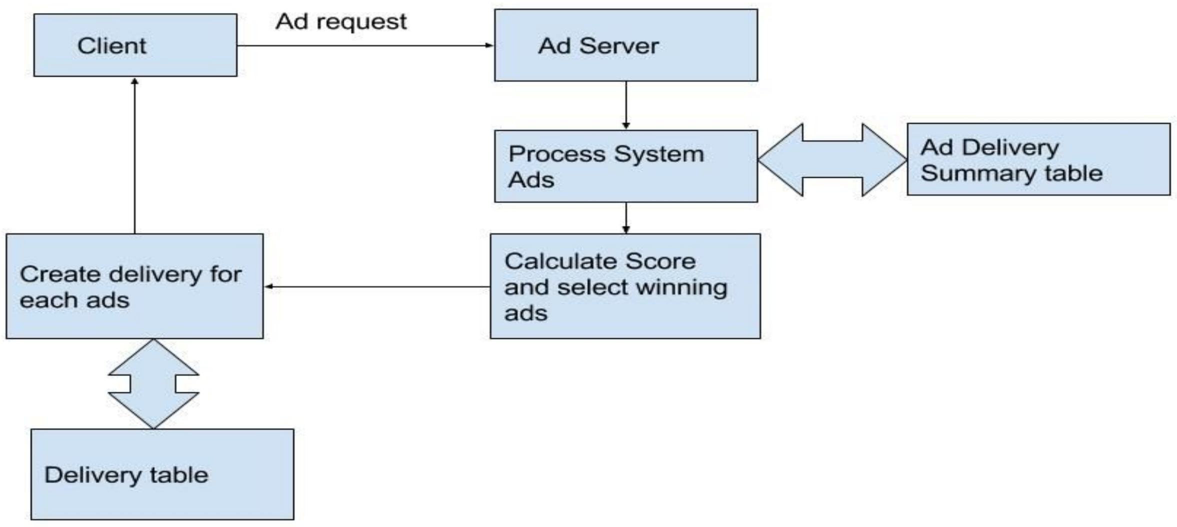
Ad Server

It will process the ad request and serve the ads to maximize the overall revenue. Here is the folder structure of the project:

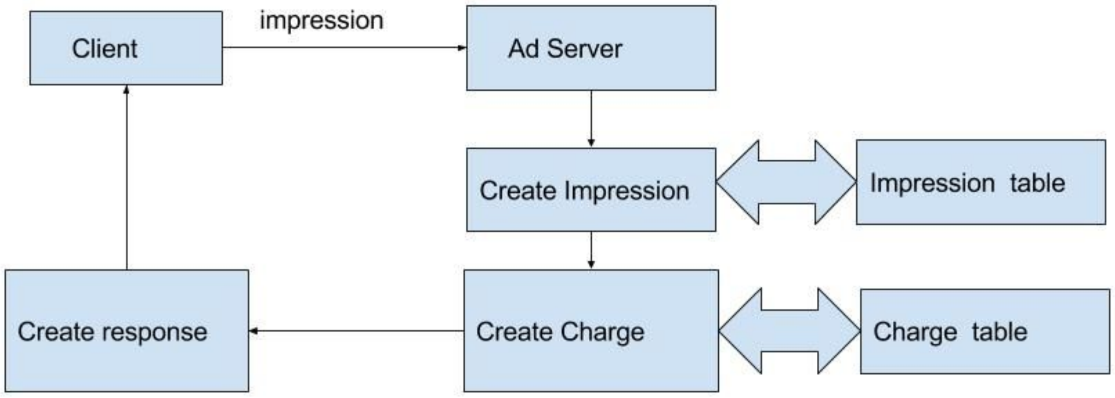
- **main.go** will bootstrap the app. This is the starting point
- **Controllers a**re responsible to process all requests and send the response to client
- **Models** are mapping of db tables
- **Services** perform all kinds of business logic
- **Helpers** provide common functionalities to all
- **Views** contain all the required templates for the app
- **Routers** provide all restful routing(LIKE GET,POST,PUT and DELETE) for the app. It's also support regex routing.
- **Requests** bind input params to a specific class
- **Responses** bind response data to a specific class

Workflow

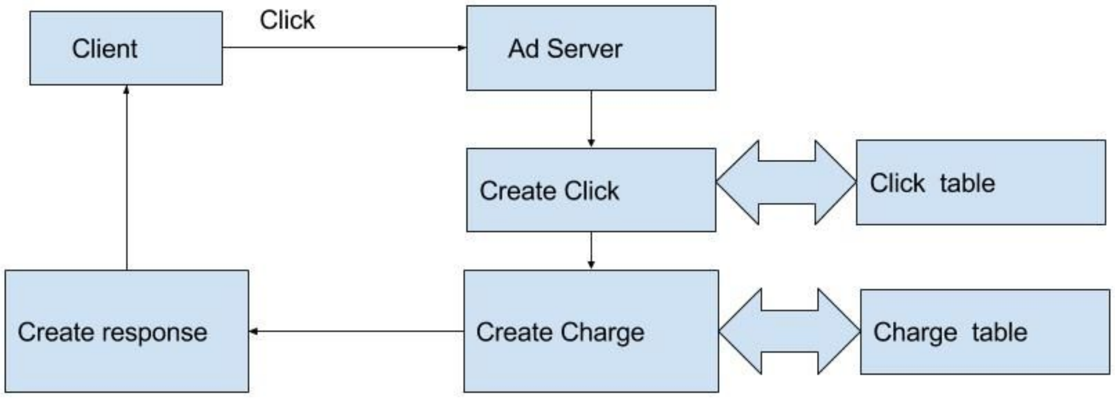
Ad request: For each ad request ad server will calculate weighted score and then will select winning ads based on the high score(Right now i am considering only CPM ads). In the database, there is a summary table(ad_delivery_summary) where we will keep stats(impression and click counter) to calculate the score quickly for all of the ads. And there will be a nightly cron that will perform cross check between summary and impression tables and will update stats accordingly.


Client
Ad request
Ad Server
Process System Ads
Ad Delivery Summary table
Create delivery for each ads
Calculate Score and select winning ads
Delivery table

Track Impression: For each impression request, ad-server will create a row into impression table and will insert charge for this impression into charge table. Right now i am diving revenue into 2 shares 60% for publisher and 40% for our service. I am storing publisher revenue in publisher_charge column.



Track Click: For each click request, ad-server will create a row into click table and will insert charge for this impression into charge table only if it is CPC ad. Right now i am diving revenue into 2 shares 60% for publisher and 40% for our service. I am storing publisher revenue in publisher_charge column.



Ad Serving Algorithm

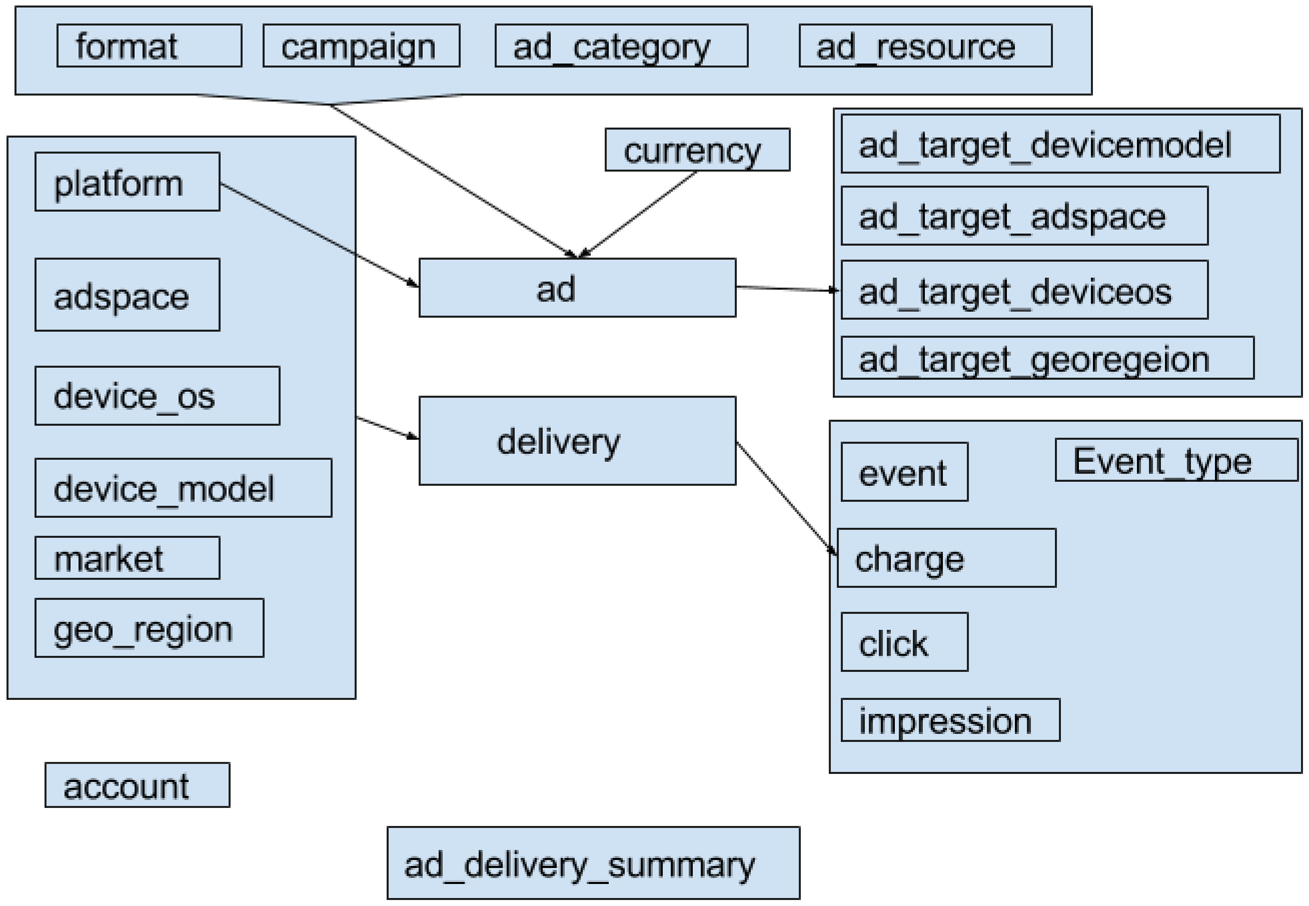
- # 1. Find all the active ads in the System. Then filter the results based on OS and Version (I.E if os=Android then it will list ads which are targeted for Android)
- # 2. Now we will have the filtered results. Initially set adPriority=0 for all ads.
- # 3. Check if there is any model target for some ads, if yes then increment the adPriority to make ad score high.(High score high probability. iOS or Android app has to sent model as param)
- # 4. Check if there is any adspace target for some ads, if yes then increment the adPriority to make ad score high.(High score high probability)
- # 5. Then check if there is any geo target for some ads. if yes then increment the adPriority to make ad score high.(High score high probability. From the user ip address we can locate the region of the ad request.)
- # 6. Now for each ad calculate weight = (maxImpressions+(adPriority*baseWeight) - servedImpressions)/(adEndtime-adStartTime) (baseWeight = 500 impresion weight)
- # 7. Then assign score for each ad, score = (weight/totalAllAdsWeight)*100
- # 8. if limit = 1 then select top 5 ads (sort the results by score desc).
- # 8.   Generate random number to select winning ad with range 1 to 5
- # 9. ELSE
- # 10.   select top n ads (order by score desc). where n=limit*2 (Results will be 2 times of the limit)
- # 11.   shuffle the results based on random number.
- #12.   then select top n results where n=Limit
- # 13.
- # 14.END
- # 15.Create delivery in delivery table for each ad
- # 16.Send the adList to the request with deliveryid

Future improvements:

- adspace historical analysis score can be added. Consider click for CPC ads
- Global cache or hashmaps to hold the weight of each ads
- Updated cache if there is any new ad
- Concurrency handler like GoRoutine
- AFTER LINE# 10:Split n ads into 2 blocks.from first block genereate random numbers to select winning ads (80% ads will served from first block and 20% from 2nd block).From 2nd block genereate random numbers to select winning ads (20% For example if Limit is 4 then 3 ads will be selected from first block and 1 //# ad will be selected for 2nd block)
- We can calibrate baseWeight value by analyzing our data.

Database Design

I have separated delivery, impression,click,event and charge into different tables to make the reports better. For example, if we want a report “region wise impression” then we can easily get that information by joining delivery and impression. In the event table we will insert custom events like MORE_BUTTON_CLICK, AD_CLOSED_BY_USER.


format
campaign
ad_category
ad_resource
platform
adspace
device_os
device_model
market
geo_region
currency
ad
delivery
ad_target_devicemodel
ad_target_adspace
ad_target_deviceos
ad_target_georegeion
event
Event_type
charge
click
impression
account
ad_delivery_summary