BETBOOK

A MOBILE APPLICATION BASED ON SWIFT

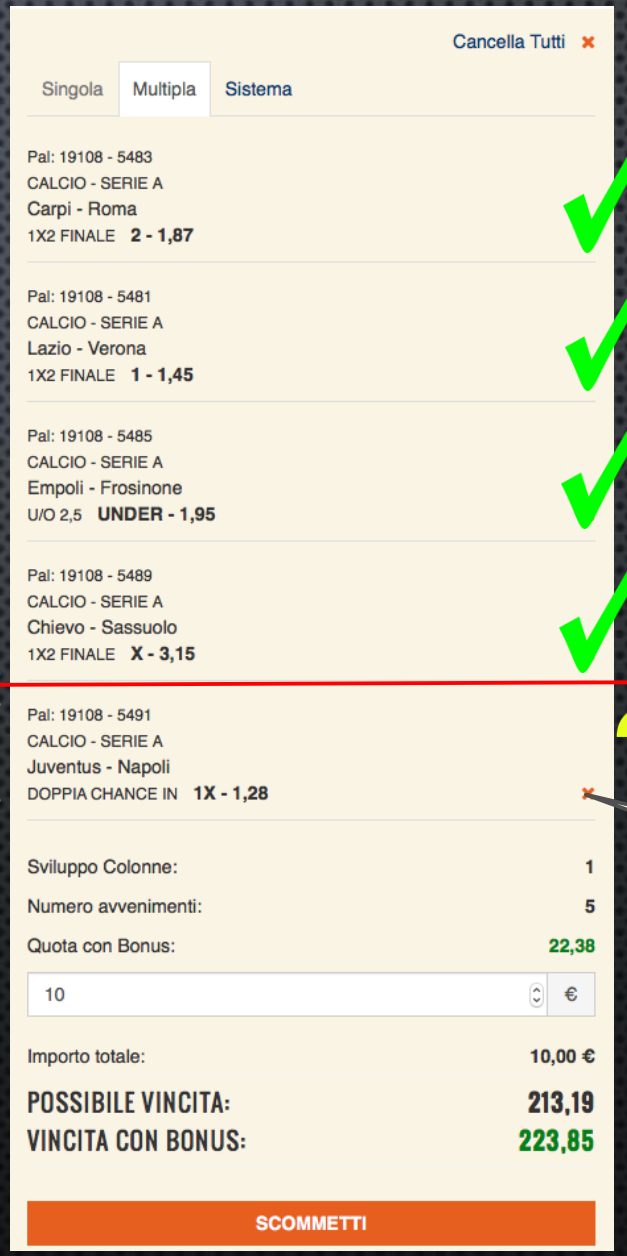
Cimbelli Alessandro
Daverio Matteo

INTRODUCTION

WHAT IT IS AND WHAT IS USED FOR

- Application to support people who want to bet on football events
- Gives information about bet coverage, quotation of teams and their performances

WHAT IT IS A BET COVERAGE



Matches that have being played and guessed

Last match to guess to win the bet!

- **You might be wondering some of the following questions…**
  - How do I cover my bet?
  - How much am I going to win? (Bounds: (Max,Min))
  - How much I need to bet on the coverage?

The answer is….Betbook!

FUNCTIONALITIES

BET COVERAGE

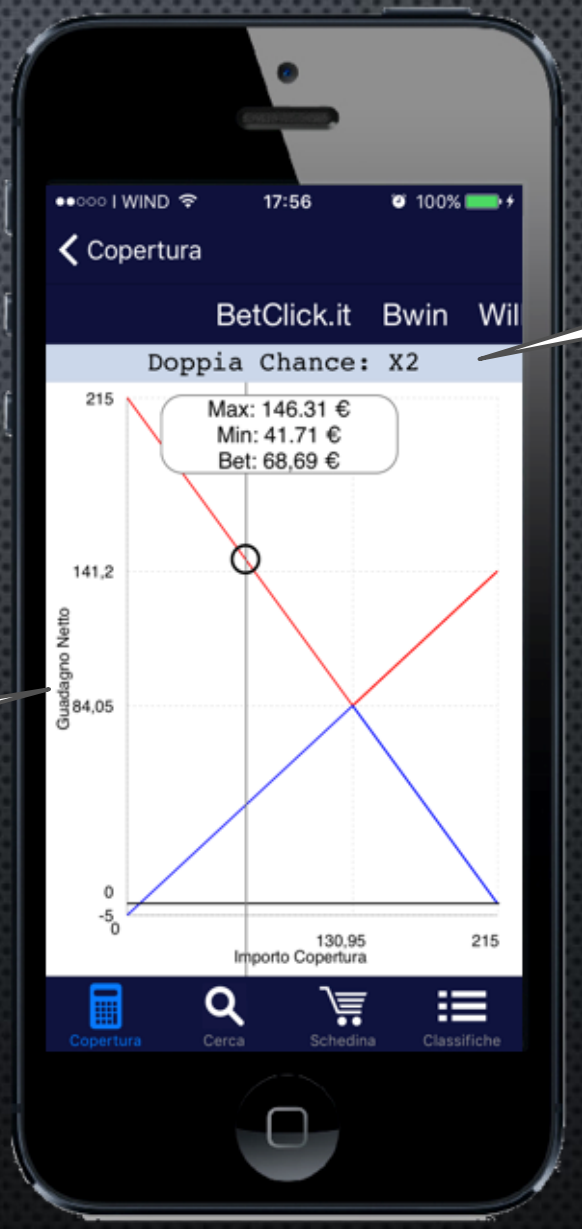
- USER CAN INSERT ALL THE DATA ABOUT HIS BET, IN A USER FRIENDLY WAY

Insert the bet using pickers

Switch to choose the modality

Button to go to the graph

- USING A CHART, THE APP SHOWS HOW USER CAN BET.
- THIS FUNCTIONALITY CAN BE USED EITHER ONLINE OR OFFLINE



In online coverage, user can see the kind of bet he has to do

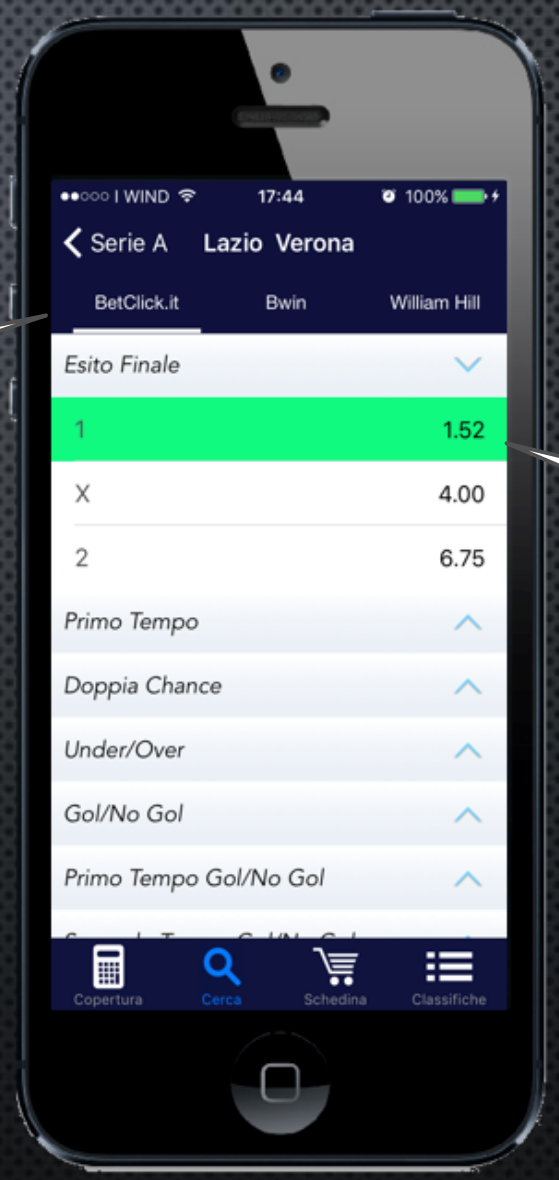
Charts display max and min win for every possible bet

EVENT VIEWER

- USER CAN SELECT THE LEAGUE THROUGH AN ACCORDION MENU AND THEN CAN LOOK AT THE EVENTS SCHEDULED

QUOTE VIEWER

- USER CAN SEE THE DETAILS FOR THE SELECTED MATCH AND CAN ALSO PUT THAT EVENT INTO THE BETTING POOL



User can change the odds' provider

Green events are in the betting pool

BETTING POOL VIEWER

- HERE USER CAN MANAGE THEIR BETTING POOL

To delete the betting pool

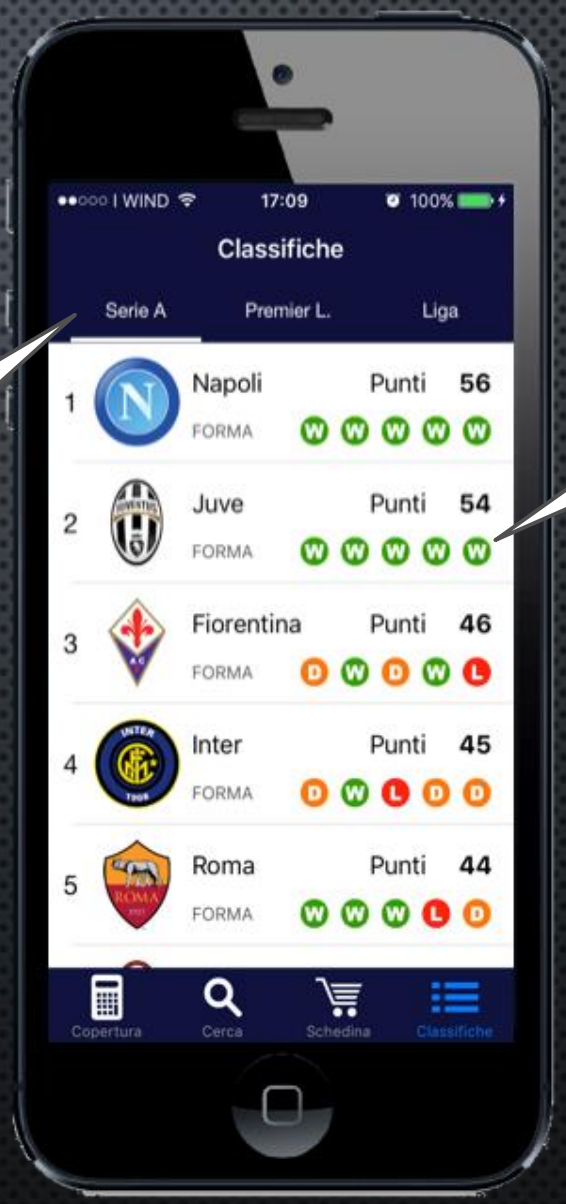
To add a new event

To delete a single event

User can modify the import and the bonus of the bet, the other data will update automatically

LEADERBOARD VIEWER

- USER CAN SEE THE LEADERBOARD WITH SOME INFORMATION ABOUT TEAMS



User can select the league

Leaderboards display the team's standings

DESIGN DECISIONS

MULTITHREADING

- **Why Multithreading?**

  Because we don't want to block the main queue's thread freezing the Ui while we do external requests. We would like to do our URL request in a different thread.

MULTITHREADING

- **EXAMPLE OF A MULTITHREADED IOS API**

  THIS API LETS YOU FETCH SOMETHING FROM AN HTTP URL TO A LOCAL FILE
  OBVIOUSLY IT CAN'T DO THAT ON THE MAIN THREAD!

```
LET SESSION = NSURLSESSION.SHAREDSESSION()
IF LET URL = NSURL(STRING: "HTTP://URL") {
    LET REQUEST = NSURLREQUEST(URL: URL)

    LET TASK = SESSION.DATATASKWITHURL(URL!){ (DATA: NSDATA?, RESPONS: NSURLRESPONSE?, ERROR: NSERROR?) -> VOID IN

        //CODE ON DATA RETRIEVED FROM THE EXTERNAL SERVICE
        SELF.DELEGATE?.SETMATCHLIST(LIST)
    }
    TASK.RESUME()
}
```

MULTITHREADING

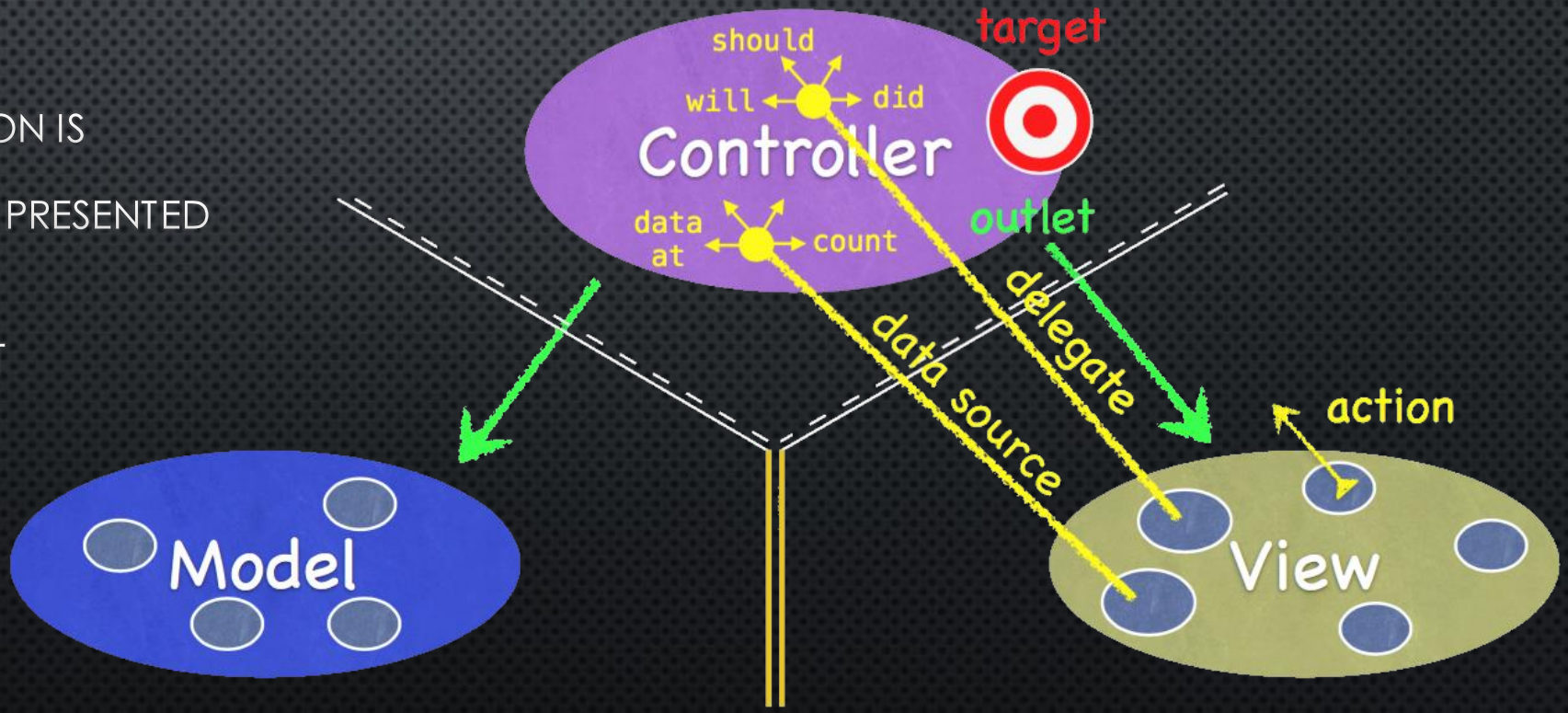
- **How to do UI stuff safely**

We simply dispatched back to the main queue the operations about UI updates

```
func setMatchList(matchList: [Match]?){
        dispatch_async(dispatch_get_main_queue()){ () -> Void in
            if( matchList == nil){
                //Display Alert Message
            }else{
                self.matches = matchList
                self.spinner.stopAnimating()
                self.refreshControl?.endRefreshing()
                self.tableView.reloadData()
            }
        }
    }
```

MODEL - VIEW - CONTROLLER

- DIVIDE CLASSES INTO 3 MAIN AREAS
- MODELS AND VIEWS NEED TO BE INDEPENDENT
- MODEL: WHAT THE APPLICATION IS
- CONTROLLER: HOW MODEL IS PRESENTED TO THE USER
- VIEW: CONTROLLER'S SERVANT



MVC: WHY WE USED IT

- It is one of the best practice for professional programming in an object oriented environment
- This pattern gave to each member of the team a common point of view on how to develop the project
- It separates business logic from UI, so code is cleaner and easier to maintain and debug
- Good reusability for model classes

Persistence

- Used to save in a persistent way betting pools
- Use of NSUserDefaults, because data are simple and small

LAYOUT

- THIS APP IS CREATED FOR ALL IPHONE DEVICES
- LAYOUT IS DESIGNED TO BE EFFICIENT ON EVERY SCREEN DIMENSIONS

Thanks for watching!

Now it's demo time.